一、

1．在计算时间复杂度时，定义：如果存在两个正常数和，对于所有的，有，则记作（ B ）。

A） B）

C） D）

2．下列排序算法属于分治算法的是（ A ）。

A）快速排序 B）希尔排序

C）基数排序 D）冒泡法排序

3．下列不属于贪心算法的是（ C ）。

A）赫夫曼树生成算法 B）最小生成树Prim算法

C）Bellman-Ford算法 D）最小生成树Kruskal算法

4．Fibonacci递归算法与动态规划算法的区别是（ A ）。

A）递归算法时间复杂度为，动态规划算法时间复杂度是。

B）动态规划算法空间复杂度大于递归算法。

C）两种算法都采用了先降低问题规模再求解的方法。

D）A、B、C都正确。

5．目前已知的NPC问题有2000多个，下面哪些不属于NPC问题（ C ）。

A）货郎问题 B）最大团问题

C）最长子序列问题 D）调度问题

6．下面程序段的中标号为的语句执行的次数是（ C ）。



A） B）

C） D）

7．下面哪些是算法的基本性质（ A）。

①目的性 ②分步性 ③有序性 ④有限性 ⑤操作性 ⑥复杂性 ⑦原子性

A）①②③④⑤ B）①②③⑤⑥

C）①③⑤⑥⑦ D）①③④⑤⑥

8．可以系统地搜索一个问题的所有解或任意解，既有系统性又有跳跃性的算法是（ D ）。

A）穷举法 B）迭代法

C）动态规划 D）回溯法

9．在一个有个元素的有序单链表中查找具有给定关键字的结点平均时间复杂度是（ B ）。

A） B）

C） D）

10．下列关于时间复杂度的选项，哪一个是由低到高排序的（ B ）。

A）

B）

C）

D）

11.算法的基本特征是（ A ）。

A）有穷性、确定性、可行性和可数的输入输出 B）正确性、可读性、规范性

C） 正确性、可行性、规范性和可数的输入输出 D）都不是

12.两个程序*f*和*g*，如果其中一个程序*f*的执行次数为n,另一个程序*g*执行次数为10n,n为循环执行次数，则（ C ）。

A）*O*(f)>*O*(g) B）*O*(f)<*O*(g)

C）*O*(f)=*O*(g) D）*O*(f)≠*O*(g)

13.分支限界算法通常有两种形式，分别为：（ D ）。

A）广度优先分支限界算法与深度优先分支限界算法

B）先进先出队列式（FIFO）分支限界算法与堆栈式分支限界算法

C）排列树法与子集树法

D）先进先出队列式（FIFO）分支限界算法与最小耗费获最大收益分支限界算法

14.递归和动态规划，它们采用的是（ D）。

A）自顶向下 B）自顶向下和自底向上

C）自底向上和自顶向下 D）自底向上

15.折半查找，其时间复杂度为：（ D ）。

A） B）

C） D）

16.回溯法和分支限界法，它们在解空间树中采用的搜索策略是（ C ）。

A）深度优先 B）广度优先

C）深度优先和广度优先 D）广度优先和深度优先

17.适应动态规划求解的问题应用具备两相基本要素：（ A ）。

A）最优子结构性和子问题重叠。 B）最优子结构性和数据存储。

C）最优子结构性和深度优先 D）子问题重叠和深度优先

18.下面程序段的中标号为的语句执行的次数是（ B ）。

|  |
| --- |
| int sum=0;  for (int i=0;i<2\*m;i++)  for (int j=0;j<n;j++)  L: sum = sum + i\*j\*2-1; |

A）  B） 2\*m\*n

C）m\*n D）

19.算法的存储量包括：（ D ）。

A）输入数据所占空间。 B）算法本身所占空间。

C）辅助变量所占空间 D）A、B、C都是

20.下列程序，变量sum打印的输出结果为：（ C ）。

|  |
| --- |
| int sum = 0;  while (sum<100)  sum = sum + 1;  printf("sum=%d",sum); |

A）0。 B）99。

C）100 D）101

二、

1．如果一个判定性问题的复杂度是该问题的一个实例的规模的多项式函数，我们称这类问题为 p 类问题 。

2．迭代算法的设计工作主要有：确定迭代模型、 建立迭代关系 式 、对迭代过程进行控制。

3．分治算法在每一层递归上都有3个步骤是分解、解决、 合并 。

4．贪心算法选择的贪心策略具有 无后效性 ，即某阶段状态一旦确定，不受这个状态以后的决策影响。

5．适应动态规划求解的问题应用具备两相基本要素： 最优子结构性质 和子问题重叠。

6．回溯法的解空间树结点类型有活结点、 扩展结点 、死结点。

7．分支限界法首先要确定一个合理的限界函数，并以此确定目标函数的界，按照 FIFO分支界限算法 或以最小耗费优先策略搜索问题的解空间树。

8．算法的空间复杂度包括：输入数据所占空间、算法本身所占空间和 辅助变量所占空间 。

9．使用 贪婪算法 的前提是“局部最优策略能导致产生全局最优解”。

10．一般情况下大整数乘法的乘数与被乘数采用 字符 类型处理。

11.算法 = 控制结构 + 原操作

12.表示算法的主要方式有： 自然语言 、 流程图、盒图、PAD图、伪代码和 程序设计语言 。

13.算法的复杂性通常包括时间复杂性和 空间复杂度 。

14.对算法的分析和评价，一般应当考虑 正确性 、可维护性、可读性、运算量、占有存储空间等诸多因素。

15.直接或者间接地调用自身的算法，称为 递归 。

16.贪心选择性质是指 以局部最优达到全局最优 。

17.回溯法在问题的解空间树中，按 深度优先 策略，从根节点出发搜索解空间树。

18.下表是一个会议的11个活动（已按结束时间递增排序），会议组希望得到活动的最大相容子集合。会议组已经安排了活动1和4，则会议组还可以安排活动 和 8,11 。

（一旦某个活动开始执行，中间不能被打断，直到其执行完毕。如果两个活动A和B不冲突，即A结束之后B才开始，或者B结束之后，A才开始，则称这两个活动A和B相容。）

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **活动** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** |
| **开始时间** | **1** | **3** | **0** | **5** | **3** | **5** | **6** | **8** | **8** | **2** | **12** |
| **结束时间** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **15** |

三、1．递归求N！算法的时间复杂度是多少，要求写出求解复杂度计算过程。

时间复杂度为O(n)

过程:

T(n)=T(n-1)+O(1)

T(n)=T(n-2)+O(1)+O(1)

=T(n-3)+ O(1)+ O(1)+ O(1)

……

= O(1)+……+ O(1)+ O(1)

= O(n)

2．算法的三要素是什么？有什么含义。

数据结构,操作,控制结构.

数据结构:算法操作的对象是数据,数据间的逻辑关系丶数据的存储方式及处理方式就是数据的数据结构.

操作:算法平台尽管有很多种类,函数库,类库也有较大差异,但必须所具备的基本操作功能是相同的.

控制结构:一个算法功能的实现不仅取决于所选用的操作,还取决于各操作之间的执行顺序,即控制结构.

3．描述贪心算法求解问题的步骤。

(1)从问题的某个解出发

(2)采用循环语句,当可以向求解目标前进一步时,就根据局部最优解策略,得到一个部分解,缩小问题的范围.

(3)将所有部分解综合起来,得到问题的最终解.

4．描述动态规划算法求解问题的步骤。

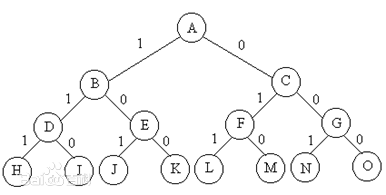
(1)分析最优子结构性质

(2)递归地定义最优值

(3)以自底向上的方式计算出最优值

(4)根据计算最优值时得到的信息,构造最优解

5．描述状态空间树，给出当时的0-1背包问题的状态空间树。



6．简述回溯法与分支限界法的区别。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 解空间搜索方式 | 存储节点的数据结构 | 节点存储特性 | 目标应用 |
| 回溯法 | 深度优先 | 栈 | 活结点的所有可行子节点被遍历后才从栈中出栈 | 找出满足条件的所有解 |
| 分支界限法 | 广度优先 | 队列,优先队列 | 每个节点只有一次成为活结点的机会 | 找出满足条件的一个解或特定最优解 |

7.请简述算法的基本步骤。

分析求解问题--🡪选择数据结构和算法设计策略--🡪描述算法--🡪证明算法正确性--🡪算法分析

8.描述NP完全问题。

这类问题中的每个问题的复杂度与整个类的复杂度有关联性,加入其中任意一个问题在多项式时间内可解的,则这类问题都是多项式时间可解.这类问题叫做NP完全问题

9.描述分治算法的框架。

Fenzhisuanfa(int n) //n为问题规模

{ if(n<=n0) //n0为可解子问题规模

{解子问题;

Return(子问题的解);}

for(i=1;i<=k;i++) //将子问题分解

yi=fenzhisuanfa(|Pi|); //递归解决Pi

T=MERGE(y1,y2,…,yk); //合并子问题

return(T);}

10.动态规划的求解步骤。

(1)分析最优子结构性质

(2)递归地定义最优值

(3)以自底向上的方式计算出最优值

(4)根据计算最优值时得到的信息,构造最优解

四、1．老板有一袋金块(共n块)，最优秀的雇员得到其中最重的一块。假设有一台比较重量的仪器，我们希望用最少的比较次数找出最重的金块。请使用伪码或者相应的平台语言实现。

maxmin(float a[],int n){

max==min=a[1];

for(i=2;i<n;i++)

if(max<a[i] max=a[i];

else if (min>a[i]) min=a[i];

}

1. 请使用伪码或者相应的平台语言实现下述功能。

写出两个整数的最小公倍数。

main(){

int x1,x2,i;

cout<<”input three numbers”;

cin>>x1>>x2;

i=1;

while(1){

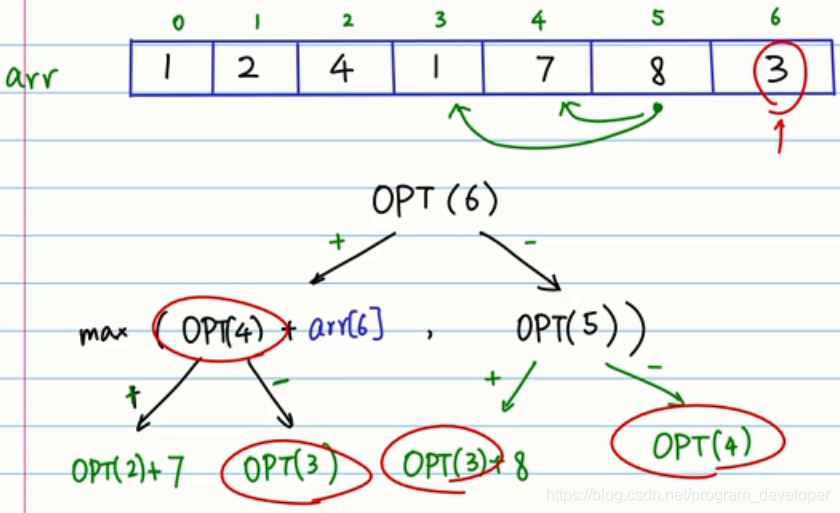
if(i mod x1=0 and I mod x2=0)break;

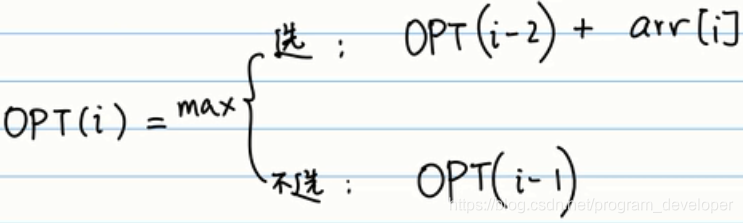
i++;}

cout<<”最小公倍数是”<<i;

}

1. 在数组中取出一个或多个不相邻数，使其和最大，即找到max(不相邻元素组成的子数组)。请使用伪码或者C/C++实现。比如数组{1，2，4，1，7，8，3}，其中的一个不相邻元素组成的子数组为{1，4， 7， 3}，其和为15。





4． 农夫john新建了个大牧场，农场中有N(2≤N≤100 000)个牛房，每个牛房都坐落在同一条线上，其坐标分别是x1，…，xN(0≤xi≤1 000 000 000)。

john有C(2≤C≤N)头牛，但这些却并不喜欢这些牛房的布局．因此，这些牛就会冲到其他牛房去．为了防止牛互相冲撞而受伤，农夫约翰就想好好安置这些牛，使它们两两之间的最小距离最大．那么这个距离到底应该是多少呢？

输入：

第一行：两个数N和C．

接下来N行，每行包含一个整数xi，代表第i个牛房的坐标。

输出：

输出一行，牛两两相距的最大的最小距离。

int a[100],n,c;

bool fun(int m){*//只要相隔m的牛的个数多余实际牛的个数，就可以返回true*

int cnt = 1,cur = 0,next=1;*//cur是当前牛编号，next是下一只牛的编号，cnt是用来计数的*

while(next<n){

next++;*//指向下一只牛*

if(a[next]-a[cur]>=m){*//当前编号牛的位置与下一编号牛的位置只要大于m*

cnt++;*//满足条件的牛的个数加1*

cur=next;*//把当前牛调整为next*

}

}

if(cnt>=c)return true;*//只要相隔m的牛的个数多余实际牛的个数，就可以返回true*

else return false;

}

int main ()

{

printf("please input the number of room:");

scanf("%d",&n);

for(int i=0;i<n;i++){

scanf("%d",&a[i]);

}

printf("input the number of cow: ");

scanf("%d",&c);

int left=1,right=(a[n-1]-a[0])/(c-1);

*//求解下界是1，就是他们紧挨着的情况，*

*//上界是最大值-最小值除（牛的个数-1），因为两头都可以取值，蓝哥思考一下为什么是牛的个数-1*

while(left<right){

int mid = ((left+right)+1)/2;

*//精髓，此处是为了确保二分法能取到最右边的数，故要让除二之前先加一，即向上取整*

if(fun(mid))left=mid;

*//此处我们需要找到满足条件的最大的值，所以如果mid点满足条件，要让left=mid，继续找更大的点*

else right=mid-1;

}

cout<<"the answer is : "<<left<<endl;

*//最后一次求得的满足条件的值mid，已经赋给了left，做一输出left*

1. return 0;
2. }

5.设计一个算法，求,要求其时间复杂度控制在常数阶（即）。

int sum = 0,n = 100;      /\* 执行一次 \*/

sum = (1 + n) \* n / 2;    /\* 执行一次 \*

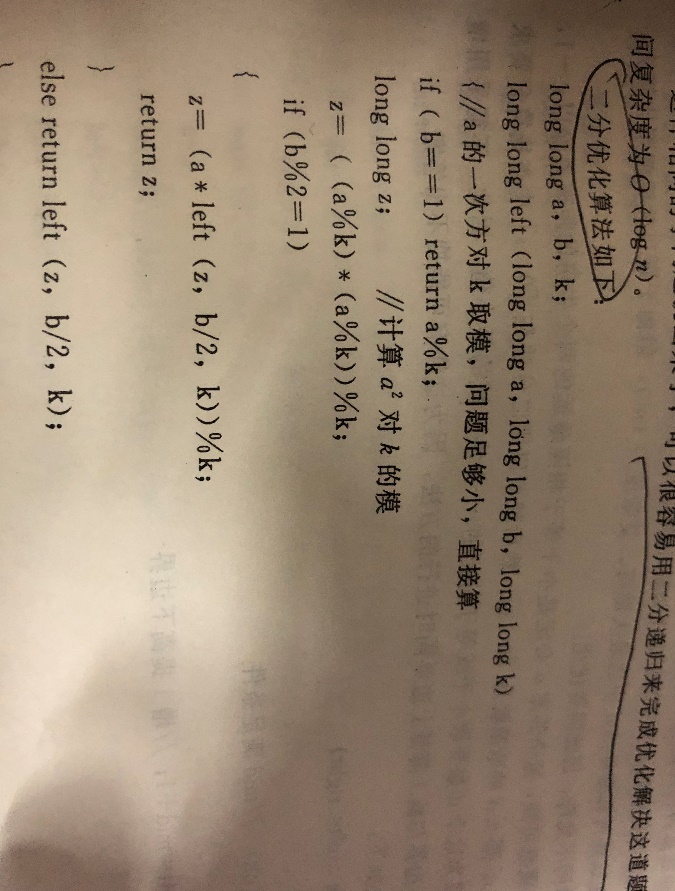
/printf("%d", sum);        /\* 执行一次 \*/

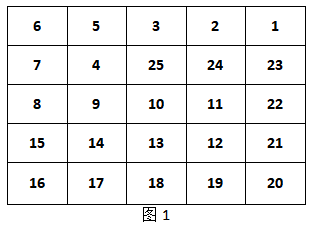
6.幂模运算是RSA加密算法的核心之一，完成以下两个问题。

（1）能不能先求，再求？如果不能，给出原因。

能

（2）设计一个算法使得时间复杂度不超过。提示。



7.Alice很喜欢滑雪，为了获得滑行速度，一定要从高处往低处滑，所以艾米必须要知道一个滑雪场最长的滑雪路径有多长。如果用数字形成的方块来表示滑雪场的区域，数字的大小代表各区域的高度，如图1所示，可以从一点滑到相连的另一点，但高度必须由高到低，相连是指互为上、下、左、右的四个方向相邻。请你设计一个算法帮Alice找出滑雪场最长的滑雪路径是哪条，有多长。

1. 以序列（-4，3，5，-5，-1，2，6，-2）为例，分别利用分治法和动态规划求解最其大子段和。采用伪代码分别写出分治法与动态规划算法的设计过程，分析两种算法的性能。
2. 请使用回溯法或者分支限定法，求解0-1背包问题。

回溯法

int Bound(int i)//限界函数:该函数返回装入所有剩余物品后(不能超过c的前提下)的价值

{

int cleft=c-cw;//剩余容量

int value=cv;

while(i<=n&&w[i]<=cleft)

{

cleft-=w[i];

value+=v[i];

++i;

}

if(i<=n) value+=v[i]\*cleft/w[i];

return value;

}

void Backtrack(int i)

{

if(i>n)//到达根节点且根节点处理完毕

{ for(int i = 1; i <= n; i++)

best\_x[i] = x[i]; //记录回溯的最优情况

bestv=cv;//更新最优的价值

return;

}

else//处理中间过程的节点

{

if(cw+w[i]<=c)//如果满足约束条件，进入左子树（约束条件：不超过容量c）

{

x[i]=1;//用来main函数构造最优解，将物品放入时x[i]=1

cw+=w[i];

cv+=v[i];

Backtrack(i+1);

cw-=w[i];

cv-=v[i];

}

if(Bound(i+1)>bestv)//满足限界函数进入右子树

{

x[i]=0;//右子树意味着物品不装入，x[i]=0

Backtrack(i+1);

}

}

}